AMENDMENTS TO THE SPECIFICATION

In paragraph [0005] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0005] Thus, to sum up, there is a need to create a non-chatty, generic, set of web methods that allow the use of simple CRUD operations for accessing directory data as a Web service, while still supporting extensibility of the directory service schema such that new directory object types can be ~~accesses~~ accessed without the need to add new directory operation methods in the client code.

In paragraph [0007] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0007] The present invention leverages the aspects of object-oriented programming and combines them with the ability of a runtime environment (e.g., the Common Language Runtime (CLR) developed by Microsoft Corporation for the .NET platform) to use metadata to properly create objects of different types to provide a simple yet flexible and extensible way to access a directory database or the like. In the system of the invention, a Web service for accessing a database works as an intermediate between clients and the database server. In accordance with the invention, the types of objects available from the database are defined by classes derived from a generic object type class, and data defining the types of the objects are included as metadata associated with the classes of the directory object types. The class definitions with the associated metadata are provided by the Web service via WSDL information to the client. When the client makes a request to perform a database operation (e.g., creating or searching for objects), the runtime library of the client uses the metadata to properly create objects of the types referred to in the request, and includes the serialized objects in the request. When the Web service receives the request, it is able to use the objects to access the database to carry out the requested database operation. With this approach, a diverse and readily extensible set of object types can be accessed, and different directory object types may be returned by the server in response to a single search. Moreover, it allows third parties to extend the directory database schema and participating in the Web services without the need to change the WebMethod signature or modifying an existing application.

In paragraph [0019] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0019] The communications between the client 90 and the Web service 92 ~~is~~ are based on a simple request-response model. To access the functionality provided by the Web service 92, the client 90 sends a request message 96 to the Web service. In the illustrated embodiment, the request message 96 is sent using SOAP over HTTP as the transport. In response, the Web service 92 returns a response message 108 to the client 90, also using SOAP over HTTP as the transport~~[AH6]~~. In alternative embodiments, other transports may be used, since SOAP may run on other transport protocols so that HTTP is not a requirement. As mentioned earlier, it is desirable to provide a simple way to enable the client to access the directory service using methods that conform to the simple CRUD model and can be used to access a set of objects of various and extensible types. As will be described in greater detail below, the approach of the invention effectively achieves this goal.

In paragraph [0022] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0022] In accordance with a feature of the invention, these issues are effectively addressed by combining the aspects of object-oriented programming with the capability of the Web services runtime mechanism to have knowledge to serialize objects to a specific type based on metadata[AH9], which may be obtained by the client via WSDL information from the server. Specifically, instead of defining a set of directory access methods for each object type found in the directory database, the various existing and future directory object types are defined as classes derived from a generic directory object class for which a simple set of directory operations (the CRUD model) are defined. The definitions of the directory object types that can be accessed by the directory operations are also included in the information provided by the Web service 92 to the client 90 to tell the client how to use its services. In a preferred embodiment, the definitions of the directory object types are in the XML format. The inclusion of the object type definitions and the use of the runtime library of the client to create instances of the objects using the definitions allow the client to pass properly constructed objects in its request to the Web service 92. This enables the Web service 92 to use the objects to interact with the directory server to carry out the intended directory operations.

In paragraph [0026] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0026] Another feature of the embodiment that should be noted is that the Search method in this example is defined such that it can return an array as the result of the search operation. This enables the search operation to produce a heterogeneous result set. In addition, the Search method takes a flexible input parameter. In one case, it takes the object itself. This allows the user to specify the filter by setting object properties. The advantage of this approach is it does not force the user to learn a certain query language or syntax such as SQL or LDAP Filter. This technique is known as Query By Example. In other ~~case~~ cases, the search takes a query string for more advanced developers. Also, the Create method is defined such that it can take an array as ~~it~~ an operand. This allows the creation of directory objects as a batch operation, i.e., sending one Create request to create multiple directory objects, which may be of different object types.

In paragraph [0028] of the originally filed application, please amend as reflected in the following marked-up version of the paragraph:

[0028] To illustrate how the directory access operations are invoked, a sample of the code [on the client is provided below.